UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/854,458	05/15/2001	Ichiro Kyushima	500.40122X00	8081

20457 7590 02/10/2005

ANTONELLI, TERRY, STOUT & KRAUS, LLP
1300 NORTH SEVENTEENTH STREET
SUITE 1800
ARLINGTON, VA 22209-9889

EXAMINER	
STEELMAN, MARY J	

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 02/10/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/854,458	KYUSHIMA ET AL.	
	Examiner	Art Unit	
	Mary J. Steelman	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *17 November 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-6 and 8-10* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-6 and 8-10* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *15 May 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This Office Action is in response to RCE, Remarks and Amendments filed 17 November 2004. Claims 1 and 8-10 are amended. Claims 1-6 & 8-10 are pending.

Claim Rejections - 35 USC § 112

2. In view of the amendments to claims 1, 8, and 10, the prior 35 U.S.C. 112 first paragraph rejections are hereby withdrawn.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-6 & 8-10 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 6,462,579 to McKinsey, in view of US Patent 6,631,518 to Bortnikov et al.

Per claim 1, McKinsey disclosed:

-A compile method in a compiler suitable for a speculation mechanism, wherein said compiler generates object codes for a processor having a speculative instruction and a speculative check instruction for checking a speculation failure (said speculative instruction and speculative check instruction are generally called a "speculation mechanism"), said compiler method comprising the steps of:

(Abstract, lines 1-8, "...method of compiling...generates object code instructions...inserting a speculation check...")

(a) performing loop-duplication during compiling operation to generate first and second loops for a repetitively executed fragment of a source program;

(A loop is a type of branch. McKinsey compiles branch code both with and without speculation (loop duplication))

(b) generating first object codes using said speculation mechanism as said first loop;

(Col. 2, lines 60-66, "...compiling source code...generates object code...inserting a speculation check into object code instructions (generates a first object code using speculation)...")

(c) generating second object codes not using said speculation mechanism as said second loop from said repetitively executed fragment of said source program;

(Col. 2, lines 65-66, "...storing recovery code (generates a second object code) associated with the speculation check..." and col. 4, lines 33-35, "The compiler...provides a recoverable interval of instructions (not using speculation) for the machine to execute in the case of failed speculation..." McKinsey generates object code to use when the speculation object code is not used.)

(d) generating third object codes that perform a control transfer of execution from said first loop to said second loop;

(Col. 4, lines 50-52, "Explicit recovery code changes the control flow (control transfer) of the optimized program. Speculating a load then implies that control flow should be added to place the recovery code." McKinsey generated code (third object code) to switch between the two types of generated object code, speculation code or non-speculation code.)

McKinsey failed to disclose a decision based on a rule of "a number of times" a speculation failure is detected by said speculative check instruction during execution of said first object codes satisfies a predetermined condition, said second object codes for said repetitively executed program fragment are executed."

However, Bortnikov disclosed profiling data for each compiled procedure. At col. 2, lines 46-53, "...organizer profile information...benchmarking phase and then, during the optimization phase, provides a system for identifying and utilizing valid profile information..." At col. 4, lines 32-33, Bortnikov disclosed "counters to be updated, accumulating branch decisions." And at col. 6, lines 34-39, "the compiler can make optimization decisions such as...when to allow early speculative execution of instructions..."

Therefore, Bortnikov suggested counters "...so that if a number of times a speculation failure is detected by said speculative check instruction during execution of said first object codes satisfies

a predetermined condition, said second object codes for said repetitively executed program fragment are executed."

It would have been obvious, to one of ordinary skill in the art at the time of the invention, to have included profile data as described in Bortnikov's invention to modify McKinsey's invention, because both inventions compile source code into variations of object code, Bortnikov additionally considers data retrieved from profiling when making a decision as to which code version to execute. Using profiling data to make informed decisions is well known in the art.

Per claim 2:

-predetermined condition in said step (c) is that the number of times a speculation failure is detected exceeds a predetermined value. (McKinsey: Col. 4, lines 58-61 and col. 6, lines 34-39.)

Per claim 3:

-predetermined condition is said step (c) is that a ratio of the number of times a speculation failure is detected by the speculation check to a number of times the repetitively executed program fragment is executed exceeds a predetermined value. (McKinsey: Col. 4, lines 58-61 and col. 6, lines 34-39.)

Per claim 4:

-when a speculation failure is detected by the speculation check, a value of counter is incremented and when the counter value exceeds a predetermined value, said third object codes transfer control to execution of said second object codes. (McKinsey: Col. 4, lines 58-61 and col. 6, lines 34-39.)

Per claim 5:

-once said speculation failure is detected, said third object codes transfer control to execution of said second object codes. (McKinsey, col. 4, lines 33-35, "The compiler of system 10 provides a recoverable interval of instructions for the machine to execute in the case of failed speculation.")

Per claim 6:

-a compiler program, tangibly stored on a computer readable medium, using said compile method according to claim 1. (McKinsey: See fig. 1 and col. 4, line 4, "compiler system 10".)

Per claim 8:

-A compile method / A computer for generating an object program... / An object program, tangibly embodied on a storage medium... for generating an object program from a source program including repetitive loop processing ,said compile method comprising the steps of:

(Abstract: "A system and method...", col. 14, lines 16-47, "A computer-readable medium...", col. 2, lines 63-64, "Object code instructions are scheduled by inserting a speculation check into the object code instructions, storing recovery code associated...and generating a control flow

graph.." Speculation code is used to optimize branch instructions. Loops are a type of branch instruction.

-generating first object codes from said source program by using a speculative instruction and a speculative check instruction for checking a speculation failure; (Col. 2, lines 60-66, "...compiling source code...generates object code...inserting a speculation check into object code instructions (first object code using speculation)...")

-performing loop-duplication during compiling operation to generate first and second loops for a repetitively executed fragment of a source program;

(Col. 2, lines65, "storing recovery code (generate a second loop, non speculation code)..." The compiler generates speculation code and recovery code for a branch / loop (loop duplication))

-generating first object codes from said source program by using a speculative instruction and a speculative check instruction for checking a speculation failure as said first loop;

(Col. 2, lines 63-65, Object code instructions are scheduled by inserting a speculation check)for checking failure) into the object code instructions (first object codes / speculation code)..."

-generating second object codes from said source program without using said speculative instruction and said speculative check instruction as said second loop;

(Col. 2, lines 65-66, "...storing recovery code (generating second object codes) associated with the speculation check..." and col. 4, lines 33-35, "The compiler...provides a recoverable interval of instructions (not using speculation) for the machine to execute in the case of failed speculation...")

-generating third object codes that perform control to first execute said first object codes;

(Col. 6, lines 47-48, "By attaching pseudo instructions representing recovery code behavior (control)...", Col. 7, line 66-col. 8, line 3, "...recovery use instructions are never scheduled explicitly. That is, the candidate selection mechanism never picks these to be scheduled. They are scheduled merely as a side effect of scheduling a recovery check instruction." In other words, the speculation code is executed first, with control switching to the non-speculation code upon failure.)

-generating fifth object codes that perform control of execution from said first loop to said second loop to execute said second object codes...

(Col. 4, lines 50-52, "Explicit recovery code changes the control flow (control transfer from speculated, first object code / loop to second non-speculated, object code / loop) of the optimized program. Speculating a load then implies that control flow should be added to place the recovery code (execute non-speculative code).")

McKinsey failed to disclose a decision based on a rule: "a number of times" / "after the number of times reaches a predetermined value". A speculation failure is detected by said speculative check instruction during execution of said first object codes that satisfies a predetermined condition, thereby directing said second object codes for said repetitively executed program fragment to be executed."

However, Bortnikov disclosed profiling data for each compiled procedure. At col. 2, lines 46-53, "...organizer profile information...benchmarking phase and then, during the optimization phase, provides a system for identifying and utilizing valid profile information..." At col. 4, lines 32-33, Bortnikov disclosed "counters to be updated, accumulating branch decisions."

Therefore, it would have been obvious, to one of ordinary skill in the art at the time of the invention, to have included profile data as described in Bortnikov's invention to modify McKinsey's invention, because both inventions compile source code into variations of object code, Bortnikov additionally considers data retrieved from profiling when making a decision as to which code version to execute. Using profiling data to make informed decisions is well known in the art.

Per claim 9:

(See rejection of claim 8 above.)

-a memory device to store said source program; a central processing unit (CPU) to execute a compiler program for generating said object program from said source program; a display device to output a result of compile processing executed by said CPU; a bus to connect said memory device, said CPU and said display device;

(McKinsey: See Fig. 1 and col. 4, lines 5-16, "...processor 12...connected (bus) to memory 16...display screen (line 15))

-wherein said CPU generates said object program by executing a compiler program that includes the steps of:

(Col. 2, lines 61-62, "generates object code instructions...")

-performing loop-duplication during compiling operation to generate first and second loops for a repetitively executed fragment of a source program;

(As noted in the rejection of claim 1, McKinsey compiles speculated code and non-speculated code. Code speculation is used with branch instructions. Loops are a type of branch instruction.)

-generating first object codes from said source program by using a speculative instruction and a speculative check instruction for checking a speculation failure as said first loop;

(Col. 2, lines 60-66.)

-generating second object codes from said source program without using said speculative instruction and said speculative check instruction as said second loop;
(Col. 2, lines 65-66 and col. 4, lines 33-35.)

-generating third object codes that perform control to first execute said first object codes;
(Col. 7, line 66- col. 8, line 3.)

-generating fifth object codes that perform control to execute said second object codes after the number of times reaches a predetermined value. (Col. 4, lines 50-52.)

McKinsey failed to disclose a decision based on a rule of "a number of times" / "after the number of times reaches a predetermined value". A speculation failure detected by said speculative check instruction during execution of said first object codes satisfies a predetermined condition, directing said second object codes for said repetitively executed program fragment to be executed.

However, Bortnikov disclosed profiling data for each compiled procedure. At col. 2, lines 46-53, "...organizer profile information...benchmarking phase and then, during the optimization phase, provides a system for identifying and utilizing valid profile information..." At col. 4, lines 32-33, Bortnikov disclosed "counters to be updated, accumulating branch decisions."

Therefore, it would have been obvious, to one of ordinary skill in the art at the time of the invention, to have included profile data as described in Bortnikov's invention to modify McKinsey's invention, because both inventions compile source code into variations of object code, Bortnikov additionally considers data retrieved from profiling when making a decision as to which code version to execute. Using profiling data to make informed decisions is well known in the art.

Per claim 10:

(See rejection of claim 8 above.)

-An object program, tangibly embodied on a storage medium... (Col. 14, lines 16-47.)

-a first object code portion generated from said source program by using a speculative instruction and a speculative check instruction for checking a speculation failure as said first loop;

(Col. 2, lines 60-66.)

-a second object code portion generated from said source program without using said speculative instruction and said speculative check instruction as said second loop;

(Col. 2, lines 65-66 and col. 4, lines 33-35.)

-a third object code portion that performs control to first execute said first object code portion;

(Col. 7, line 66- col. 8, line 3.)

-a fifth object code portion that performs control of execution from said first loop to said second loop to execute said second object code portion...

(Col. 4, lines 50-52.)

McKinsey failed to disclose a decision based on a rule of "a number of times" and "after the number of times reaches a predetermined value". A speculation failure is detected by said speculative check instruction during execution of said first object codes which satisfies a predetermined condition, directing said second object codes for said repetitively executed program fragment to be are executed.

However, Bortnikov disclosed profiling data for each compiled procedure. At col. 2, lines 46-53, "...organizer profile information...benchmarking phase and then, during the optimization phase, provides a system for identifying and utilizing valid profile information..." At col. 4, lines 32-33, Bortnikov disclosed "counters to be updated, accumulating branch decisions."

Therefore, it would have been obvious, to one of ordinary skill in the art at the time of the invention, to have included profile data as described in Bortnikov's invention to modify McKinsey's invention, because both inventions compile source code into variations of object code, Bortnikov additionally considers data retrieved from profiling when making a decision as to which code version to execute. Using profiling data to make informed decisions is well known in the art.

Response to Arguments

5. Applicant's arguments filed 17 November 2004 have been fully considered but they are not persuasive.

Applicant has argued, in substance, the following:

(A)

As Applicant has pointed out on page 7, last paragraph – page 8, first paragraph, of RCE / Remarks, received 17 November 2004, "...neither McKinsey nor Bortnikov et al., disclose, "suggest or render obvious the limitations in the combination of each of these claims of, inter alia, performing loop duplication during compiling operation to generate first and second loops for a repetitively executed fragment of a source program, or generating object codes that perform a control transfer of execution from the first loop to the second loop based on a predetermined condition or value."

Examiner's Response:

See rejection of claim 1 above. McKinsey does provide for "generating first object codes using said speculation...", "generating second object codes not using said speculation (recovery code)..., and "generating third object codes that perform a control transfer..." that are used if recovery code is to be executed. See "SUMMARY OF THE INVENTION", col. 2, line 58 – col. 3, lines 6. The Bortnikov reference is used to provide rejection for detecting a number of times a speculation fails. Bortnikov disclosed utilizing profile information, including counters for branch decisions (col. 4, lines 32-33) to decide when to allow early speculation (col. 6, lines 34-

30. Bortnikov did disclose (col. 4, lines 54-61), "Various methods of optimizing program code with profile data...", whereby various phases are performed, including an optimization phase where the program is recompiled and optimized in light of the profile information.

Recovery code, well known in the art, is compiled code used when speculated branch code is not taken (i.e., speculation failure). A loop is a type of branch. As disclosed by McKinsey, the generated speculation code for a loop is a 'first object code'. The related, generated **recovery code** for a loop or 'loop duplication', creates a 'second object code'. The compiler generates both speculation code and recovery code, thereby duplication of the loop code. The compiler provides control flow (object code to transfer control) to transfer execution between said 'first object codes / first loop' and said 'second object codes / second loop'.

(B) As Applicant has pointed out on page 8, first paragraph "McKinsey merely disclosed compiling source code where a compiler generates immediate code from the source code, generates object code instructions from the intermediate code and schedules the object code instructions" and fails to disclose "generating first object codes using speculation mechanism as a first loop, generating second object codes not using the speculation mechanism as the second loop, or generating object codes that perform a control transfer of execution from the first loop to the second loop."

Examiner's Response:

Examiner strongly disagrees. McKinsey disclosed (col. 4, lines 1-3) "...creating and updating control and data speculative recovery code during each phase of compilation...", (col. 4, line 28), "...machine code (object code) is generated...", col. 4, lines 33-35, "The compiler of system provides a recoverable interval of instructions (second object codes) for the machine to execute in the case of failed speculation." Col. 4, lines 52-53, "Speculating a load (speculation mechanism - using first object codes) then implies that control flow (control transfer) should be added to place the recovery code (not using speculation mechanism - second object codes).

(C) As Applicant has pointed out on page 8, first paragraph, Bortnikov et al. merely discloses a profiling system wherein profile data is stored in a separable hierarchical fashion such that profile data for each compile procedure in a computer program can be readily identified and utilized."

Examiner's Response:

Examiner strongly disagrees. Bortnikov disclosed profilers that insert instructions that count. Collected information can be used to optimize. Col. 5, lines 44-45, "compilers can automatically insert instrumentation code into the created object modules during the compilation process." Col. 6, lines 23-25, "The instrumentation code will typically include access to control flow counters that will be incremented each time (fourth object codes to count a number of times...)...", col. 6, lines 34-39, "the compiler can make optimization decisions such as...when to allow early speculative execution of instructions (speculation mechanism – using first object codes);

Claim limitations call for generating object code for a speculative mode, a non speculative mode and control to transfer between execution of the two object codes. Counters are instrumented into the code to limit certain control paths.

Examiner maintains the rejections of claims 1-6 & 8-10.

Conclusion

6. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Note related prior art:

US Patent 5,202,995 to O'Brien: Col. 3, lines 50-66, "loop that has an invariant conditional branch instruction is transformed into two loops...The conditions of the original conditional branch instruction may be evaluated before either loop is entered. If this condition evaluates false, then the first newly created loop is executed. Otherwise, the second newly created loop is executed."

US Patent 5,761,515 to Barton, III et al: Col. 7, lines 3-5, "enables the compiler to generate two sequences, one optimized assuming cache-hit, and the other optimized assuming cache-miss. (generate two object codes, use depends on value of counter)"

US Patent 6,151,706 to Lo et al: Col. 3, lines 45-48, "profile-driven speculation heuristic is used...allows the amount of speculation to be controlled..."

US Patent 6640315 B1 to Hwu et al: Abstract, lines 1-2, "Disclosed is a method and system for handling inline recovery from speculatively executed instructions", col. 6, lines 21-22, "indicates whether the currently executing instruction should be executed under normal semantics or recovery semantics.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached at (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman 

02/01/2005

WEI Y. ZHEN
PRIMARY EXAMINER